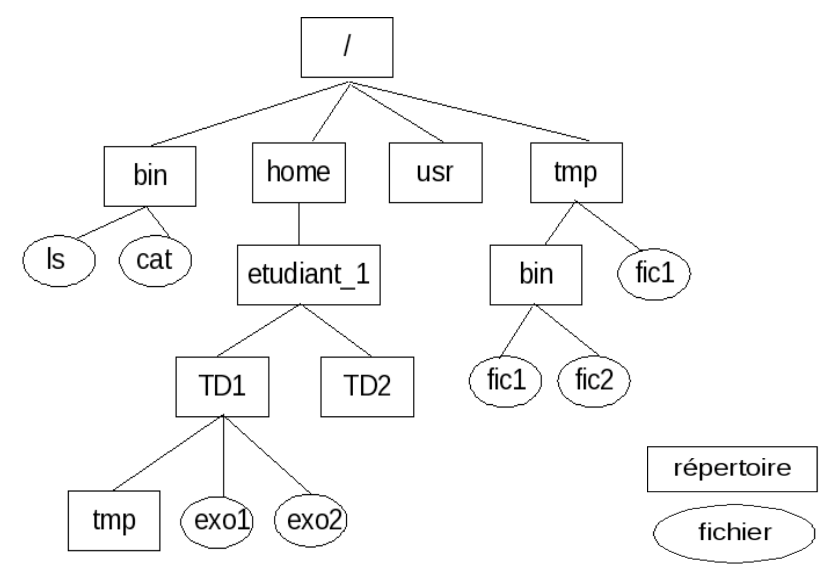
Aide mémoire UNIX

Le système de fichiers

Les informations (données, programmes, images, courriers...) sont stockées dans des fichiers (file) qui sont eux-mêmes enregistrés dans des répertoires (directory).
Sous UNIX la structure des données, ou système de fichiers se présente sous la forme d'un arbre inversé contenant :

- un point d'entrée appelé racine (root) et noté /
- des répertoires (noeuds) pouvant contenir d'autres noeuds et/ou des feuilles
- des fichiers qui constituent les feuilles de l'arbre

Exemple d'arborescence



Les droits d'accès aux fichiers
UNIX est un système multi-utilisateur. Afin que les données de chacun soient à l'abri des autres utilisateurs, chaque fichier appartient à un utilisateur précis (en général son créateur) et à un groupe d'utilisateurs.
De plus, à chaque fichier est attribué un ensemble de droits d'accès qui concernent :

- le propriétaire du fichier,
- le groupe à qui appartient le fichier,
- l'ensemble des utilisateurs de la machine.

Pour chacune de ces trois catégories correspond un droit de lecture, d'écriture et d'exécution.

- Le droit de lecture donne le droit de lire le fichier.
- Le droit d'écriture de le modifier, ou de l'effacer.
- Le droit d'exécution permet d'exécuter le fichier si celui-ci contient un programme.
- Le droit d'exécution donné à un répertoire indique que l'on a le droit de le parcourir.

Ainsi, le plus souvent le propriétaire du fichier a les droits de lecture et d'écriture, voire au besoin d'exécution, alors que le groupe à juste un droit de lecture et éventuellement d'exécution, ce qui permet de partager son travail au sein d'une équipe. Enfin le reste des utilisateurs a soit des droits similaires au groupe soit aucun droit.
Ces droits peuvent être modifiés par le propriétaire du fichier.

La dénomination des fichiers et la notion de chemin (path)

Sous UNIX les noms de fichier peuvent contenir :

1. des caractères alphabétiques (a-z et A-Z)
2. des caractères numériques (0-9)
3. des signes de ponctuation (&$+=*....)

Cependant l'utilisation de certains signes peut poser problème et il est recommandé de se limiter aux **"_"** et **". "** (en général suivit de l'extension).

Attention :
1. ne jamais mettre de caractère espace dans le nom d'un fichier
2. pas utiliser de caractères accentués
3. les majuscules et les minuscules sont des caractères différents, ainsi les noms TOTO, toto, Toto et ToTo sont tous différents et désignent des fichiers différents.

La dénomination d'un fichier peut contenir les indications sur la **localisation** de celui-ci dans l'arborescence.
Dans ce cas, le nom du fichier est précédé du **chemin** (**path**) qui mène à lui.
Chaque étape de ce chemin est séparée par le caractère /

exemple : /tmp/fic1
désigne dans l'arborescence précédente le fichier fic1, localisé dans le répertoire tmp, lui même situé sous la racine

remarques :
- **si le chemin commence par le caractère /**
  le chemin est dit **absolu** et la première étape de celui-ci est située dans le répertoire racine. Dans l'exemple ci-dessus, le répertoire tmp est effectivement placé dans le répertoire racine.

- **si le chemin ne commence pas par le caractère /**
  Le chemin est dit **relatif** et la première étape de celui-ci se trouve dans le répertoire **courant** (répertoire d'où est lancée la commande qui utilise le nom de fichier).

- **si le chemin commence par le caractère ~**
  Le chemin est relatif au **répertoire de login (home directory)** de l'utilisateur exécutant la commande.
  Ainsi, si l'utilisateur etudiant_1, a pour répertoire de login : /home/etudiant_1
  le fichier désigné par ~/TD1/exo1 fera référence, pour cet utilisateur, au fichier exo1 localisé dans le répertoire /home/etudiant_1/TD1

raccourcis :
- **le caractère .** dans un chemin désigne le **répertoire courant**
- **les caractères ..** dans un chemin désignent l**e répertoire père** du répertoire courant, c'est à dire celui qui dans l'arborescence des fichiers se trouve juste au dessus.

Les principales commandes Shell

- **la gestion des répertoires**

pwd "print working directory"
syntaxe : pwd
Affiche le nom du répertoire courant
Permet de connaître sa position dans l'arborescence des répertoires

cd "change directory"
syntaxe : cd [destination]
Permet de se déplacer dans l'arborescence.
Le répertoire courant est modifié en fonction de la destination indiquée. Les [] autour de destination indiquent que cet argument est optionnel. Ainsi :
- Si aucune destination n'est indiquée : le répertoire courant est positionné au niveau du répertoire de login (home directory) de l'utilisateur
- Si une destination est indiquée : le répertoire courant est positionné au niveau du répertoire dont le nom est donné en destination

Il est possible d'indiquer le répertoire de destination de façon **absolue** (chemin d'accès décrit à partir de la racine) ou **relative** (chemin d'accès décrit à partir du répertoire courant).
Il est également possible d'utiliser des raccourcis tels que :

.	désignant le répertoire courant
..	désignant le répertoire père (celui au dessus dans l'arborescence)
~	désignant le répertoire de login
/	désignant la racine

exemple : **cd ..** positionne le répertoire courant au niveau du répertoire père.
cd ~ positionne le répertoire courant au niveau du répertoire de login
cd / positionne le répertoire courant au niveau de la racine

ls "list"

syntaxe : ls [-alF] [fichiers]
Liste le contenu d'un répertoire à l'écran en indiquant l'ensemble des fichiers présents ou seulement une partie d'entre eux suivant les indications spécifiées par l'argument fichier.
exemple : ls ex* listera l'ensemble des fichiers dont le nom commence par ex

Options :
- a : (all) liste l'ensemble des fichiers y compris les fichiers cachés commençant par un "."
- l : (long) affiche les informations sur les fichiers dans la forme longue, avec les informations de type, date, droit et propriétés.
-F : indique par un caractère, ajouté a la fin du nom de fichier, le type de celui-ci :
  - / : répertoire
  - * : exécutable

mkdir "make directory"

syntaxe : mkdir nom
Permet la création, dans le répertoire courant, d'un répertoire défini par l'argument nom.
Cet argument suit les règles standard de dénomination des fichiers : chemin absolu ou relatif.

rmdir "remove directory"

syntaxe : rmdir nom
Permet la destruction du répertoire désigné par l'argument nom.
Cet argument suit les règles standards de dénomination des fichiers : chemin absolu ou relatif.
Attention , pour pouvoir détruire un répertoire il faut :
1. que celui-ci soit vide de tout fichier et de tout sous-répertoire
2. être positionné dans un autre répertoire

• La gestion des fichiers :

cp "copy"

syntaxe : cp source destination
Effectue une copie du fichier désigné par source dans le fichier désigné par destination.
Vous devez avoir le droit de lecture sur le fichier source et d'écriture dans le répertoire où vous placez le fichier de destination. La syntaxe des noms de fichiers suit les règles standards de dénomination des fichiers : chemin absolu ou relatif.

mv "move"

syntaxe : mv source destination
Déplace le fichier désigné par source. Plusieurs possibilités :
1. Si destination est le nom d'un répertoire, le fichier est déplacé dans ce répertoire et son nom reste inchangé.
2. Si destination est le nom d'un fichier précédé d'un chemin d'accès (relatif ou absolu), le fichier source est déplacé et son nom est modifié.
3. Si destination est un nom de fichier non précédé d'un chemin, le fichier source est "déplacé" au sein du même répertoire, ce qui consiste simplement à le renommer.

rm "remove"

syntaxe : rm fichier
Détruit **définitivement** et **irrémédiablement** le fichier désigné.

• Consultation des fichiers :

cat

syntaxe : cat fichier
La commande cat permet d'afficher à l'écran le contenu d'un fichier.

Si le fichier est trop grand pour tenir complètement à l'écran, le début du fichier est perdu et seul les dernières lignes sont visibles.

more

syntaxe : more fichier
more permet de lire le contenu d'un fichier page par page et non d'un seul bloc comme cat.
Une pression sur la barre d'espace permet de faire défiler le texte d'une page et une pression sur la touche retour chariot (ou entrée, ou return suivant les claviers) fait avancer le texte d'une ligne.
Si le nom du fichier à traiter est absent alors more travaille avec l'entrée standard.

- ## Autres :

man "manuel"

syntaxe : man commande
Cette commande est très utile car elle permet d'obtenir le manuel d'utilisation d'une commande Shell donnée en argument

& rajouté à la fin d'une ligne de commande ce caractère a pour effet de provoquer l'exécution de la commande en tache de fond, ce qui vous permet de garder la main sur le terminal.
exemple : nedit & permet de lancer l'éditeur nedit en tache de fond